140. When the NSM detected suspicious behavior (through the analysis of anomalous behavior, anomalous data path, or signature detection) it responded in two different ways. First, the NSM could raise an alarm for the security officer. Second, the NSM could alter the analysis of collected data. *NSM 1990* describes this as a "monte carlo divide-and-conquer search" of the data collected. The idea was motivated by the concern that the computer that the NSM was running on would not have the computational power to perform anomaly detection on all objects all the time, so it would only initially perform the anomaly detection at the higher-level groupings (e.g., Source or Source-Destination objects), and the NSM would drill down and examine the more detailed objects (e.g., Source-Destination-Service or Connection objects) only if one of the larger groupings appeared suspicious. Thus, if the NSM detected anomalous behavior at the Source-Destination object, it could alter its planned analysis to also include examining the Source-Destination-Service objects underneath it.

141. *NSM 1990* also described plans to extend the approach to "hybrid systems" that involved "host-based monitors to watch over the activities of individual hosts."[82] *NSM 1990* discussed the need to extend the approach of the current implementation to "distributed monitoring of wide area networks," where the "network monitoring functions have to be distributed among several nodes" and these nodes would "exchange information to reach a consensus on whether an attack is in progress."[83]

142. In my opinion, as more fully reflected in Exhibit C to my report, *NSM 1990* as a publication satisfied every limitation of the indicated claims and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the inventions claimed in the '338 patent. Consequently, the noted claims of the '338 patent were not novel when filed.

---

[82] *NSM 1990* at 302.
[83] *NSM 1990* at 302-303.

143. By 1991 NSM was in public use. At trial I may testify regarding the use of NSM on the UC Davis campus and at several government-owned locations. In my opinion, as more fully reflected in Exhibit C to my report, this public use of NSM anticipates the indicated asserted claims of the patents-in-suit.

144. In addition, as reflect in Exhibit C, all of the rest of the asserted claims of the '338 patent are invalid due to obviousness.

145. As described elsewhere in my report, it would have been obvious to one of ordinary skill in the art to combine NSM with other intrusion detection projects—including DIDS, GrIDS, and ISM—developed in whole or in part by researchers at UC Davis. In fact, NSM was part of both the DIDS and ISM architectures, and documentation associated with GrIDS explicitly describes using GrIDS in conjunction with NSM.[84] One of skill would have been motivated to combine these related systems, all of which were directed at detecting computer intrusions, to improve the functionality of the overall system.

146. It also would have been obvious to one of ordinary skill in the art to use the statistical profile-based anomaly detection of network traffic data practiced by NSM to analyze the additional network traffic data categories analyzed and described by: "A Network Security Monitor—Final Report," NetRanger, ISS RealSecure, Network Flight Recorder, Network Security Probe, Gabriel, synkill, SNMP/RMON IETF standards, and the SunScreen Firewall.[85] As shown in these references, all of these network traffic data categories were well-known, and it was also well-known that these categories could be analyzed using either signature detection, statistical profile-based anomaly detection, or both. One of skill would have been motivated to combine NSM with these additional

---

[84] *See* GrIDS webpage discussing using GrIDS with NSM. [SYM_P_0512095, 2096]
[85] *See* the NSM chart at Ex. C for the full titles and dates of all these references.

references in order to monitor additional types of network traffic for improved detection capabilities.

147. Furthermore, as previously discussed, it would also have been obvious to use the NIDES algorithms to implement the statistical profile-based anomaly detection practice by NSM. Indeed, *NSM 1990* explicitly suggests "looking at generating masks using the techniques used by IDES, Wisdom and Sense, and other intrusion detection systems, so that we can make an experimental analysis of these different methods."[86] One of skill would have been motivated to combine NSM with the NIDES algorithms based upon this explicit suggestion.

148. It would have been obvious to one of ordinary skill in the art to use the statistical profile-based anomaly detection of network traffic data practiced by NSM with distributed, hierarchical intrusion detection systems like DIDS, NetRanger, ISS RealSecure, CIDF, AIS, ISM, GrIDS, and Emerald 1997. *NSM 1990* explicitly suggests extending NSM to a distributed environment.[87] One of skill would have been motivated to combine NSM with these additional systems because both NSM and these other systems were designed for detecting computer intrusions, and NSM explicitly suggested such a combination.

149. In addition, to the extent there were any differences between the configuration of NSM and the configuration described in the patents-in-suit, it would have been obvious to modify the configuration of NSM based upon the nature of the problem to be solved. Such configuration changes would have been motivated by a desire to address the problem of detecting and responding to suspicious network activity

---

[86] *NSM 1990* at 299.
[87] *See NSM 1990* at 302-03. Indeed, several designers of distributed intrusion detection systems either used or expressed an interest in using NSM within such a system. *See* discussion of DIDS, ISM and GrIDS. *See also Emerald 1997* at 364.

in very large enterprise networks. For example, as the NSM's primary developer, I went on to develop Network Radar, which did in fact address many of these differences.

**2. DIDS**

150. The Distributed Intrusion Detection System (DIDS) was a combined network and host-based intrusion detection system first developed in the early 1990s at UC Davis, LLNL, and Haystack Labs. Many different publications have described various aspects of the DIDS system, as listed in Exhibit D. This printed publication portion of this report focuses on DIDS as it was described in: Steven Snapp et al., "Intrusion Detection Systems (IDS): A Survey of Existing Systems and A Proposed Distributed IDS Architecture" (February 1991) ("*DIDS Feb. 1991*") and S.R. Snapp, J. Brentano, G.V. Dias, L.T. Heberlein, C. Ho, K.N. Levitt, B. Mukherjee, (with S.E. Smaha, T. Grance, D.M. Teal, D.L. Mansur), "DIDS -- Motivation, Architecture, and an Early Prototype" Proc. 14th National Computer Security Conference, Washington, DC, Oct. 1991, pp. 167-176 ("*DIDS Oct. 1991*").

151. DIDS was a two-level, heterogeneous intrusion detection system, as shown in Figure 3.[88] At the first level, traditional network-based and host intrusion detection systems detected suspicious activity for the subjects that they monitored. These monitors used combinations of methods such as misuse and anomaly detection to detect suspicious activity. At the second level, the DIDS Director correlated activity across the network using a construct called the Network Identifier, or NID. The Director used the NID to aggregate/integrate reports of suspicious activity from the multiple host and LAN monitors. The Director could respond to suspicious activity by alerting a security officer

[88] As the inventors themselves have stated "[f]urther research by UC Davis in the Distributed Intrusion Detection System (DIDS) [23] and later Graph-based Intrusion Detection System (GRIDS) [24] projects has attempted to extend intrusion monitoring capabilities beyond LAN analysis, to provide multi-LAN and very large-scale network coverage." *See* Live Traffic Analysis at p. 3.

or directing the individual host and LAN monitors to modify their analysis. Finally, the papers documented future plans for DIDS including monitors for network gateways and file servers, and additional levels of hierarchy.

152. The NSM was the network monitor being used in the DIDS system. *DIDS Oct. 1991* explained that NSM was the LAN monitor being used and explicitly mentioned profiles, and further directed the reader to *NSM 1990.*[89] *DIDS Feb. 1991* also explained that NSM was the LAN manager, and again directed the reader to *NSM 1990.*[90] Given this explicit direction to use *NSM 1990* to build the DIDS LAN monitor, *NSM 1990* is incorporated into the text of each DIDS publication. In the alternative, the combination of the two is obvious, as it is explicitly suggested by the references themselves. The actual DIDS implementation contained NSM and all of its anomaly detection capabilities.[91]

153. As documented in *DIDS Feb. 1991* and *DIDS Oct. 1991*, DIDS had the following salient features:

- Performed intrusion detection across multiple monitors (hence the term "Distributed" in "Distributed Intrusion Detection System").
- Performed two-level hierarchical monitoring, with LAN and host monitors at the first level and the Director at the second level.
- Correlated activity across multiple hosts and network connections via the Network Identifier (NID).
- Aggregated suspicious activity across multiple users, hosts, and network connections.

---

[89] *DIDS Oct. 1991* at 167, 175.
[90] *DIDS Feb. 1991* at 1, 15.
[91] *See* Terrance Lee Goan Jr., "Towards a Dynamic System for Accountability and Intrusion Detection in a Networked Environment," M.S. Thesis, Division of Computer Science, University of California, Davis, 1992 at 35.

- Coordinated multiple host monitors (based on Haystack) and network monitors (based on NSM).
- Detected suspicious activity on hosts and in network connections using signature and anomaly detection.
- Designed to support multiple network monitors and multiple host monitors spanning multiple LANs.

154. DIDS was designed to aggregate all potentially suspicious activity across a network performed by a single user to that one user, no matter how many different hosts, user names, and network connections were used by that individual. Previously, host-based intrusion detection systems largely analyzed actions by individual user names independently from other user names on the same system or even the same user name on different hosts. Similarly, network-based intrusion detection systems largely analyzed connections independently from other connections. For example, if there was a connection from host A to host B and another connection from host B to host C, a network-based intrusion detection system would analyze these connections independently.

155. DIDS LAN monitors (NSMs) monitored the same categories of network traffic discussed previously in numerous NSM publications. These categories were thus already known to one of skill reading the DIDS publications. There was no need in the DIDS publications to reinvent the wheel. Furthermore, as shown in Exhibit D, both DIDS references explicitly discuss monitoring multiple different categories of network traffic, such as network connections and network packet data volume.

156. Figure 4 demonstrates the problem DIDS addressed. A user (1) sits down at Host1 and logs in as user Bob (2). Eventually the user connects to Host2 (3) and logs in as Eve (4). The user also connects to Host3 (5) and logs in as user Bob (6). At some point the user executes a "substitute user" (su) command (7) to become user Mary (8) on

Host3. The user, from the Mary account on Host3, tries to connect to Host4 (9) and Host5 (10). The user, from the Eve account on Host2, also tries to connect to Host6 (11).

157. Prior to DIDS, a typical collection of host-based intrusion detection systems would report activity from four different users on three different machines (plus perhaps activity from hosts 4, 5, and 6). Meanwhile, a typical network-based intrusion detection system would report five different connections. DIDS first goal was to perform correlation to recognize that these nine activities originate from an initial login point. DIDS second goal was to aggregate all the suspicious activity from these activities to this initial login point.

158. DIDS performed correlation by having host and network-based sensors report low-level audit events associated with logins, user name changes, and network activity to a centralized sensor called the DIDS Director. The DIDS Director had a rule-based system that used each of these low-level audit events to construct a single Network Identifier (NID). The NID was the mechanism of correlation. DIDS performed aggregation/integration by having each of the host and network-based sensors report suspicious activity to the DIDS Director. Because the DIDS Director could map activity from each user on each machine and each network connection to a single NID, any and all suspicious activity reports associated with the same NID could be aggregated together. This allowed activity that might be only slightly suspicious at any point in the network to be aggregated together and become clearly suspicious. The correlation of activity across the network via the construction of the NID was a primary focus of all the DIDS papers.

159. As shown in Figure 5, DIDS had three types of monitors - host monitors, LAN monitors, and a Director - and each had its own data sources. The host monitor processed audit records generated by the host's operating system (1). Similarly, the LAN monitor processed packets read from the network (1). Both the host and LAN monitors generated two types of data processed by the Director: low-level events (2) and reports

of suspicious behaviors (3). The low-level events (2) consisted of events such as user logins from the host monitor and connection starts from the LAN monitor, and the Director used these low-level events to construct the NID. The host and LAN monitors also generated reports of suspicious behaviors for the subjects they measured (e.g., user activity for the host monitors and connections for the LAN monitors), and the Director aggregated these suspicious reports together to arrive at an aggregate suspicion measure for the NID.

160. The host and LAN monitors in DIDS used well-known methods for determining if the subjects they monitored were behaving suspiciously or not. These methods included misuse detection (e.g., using signatures) and anomaly detection (e.g., using profiles). The Director used a rule-based system to correlate the low-level events into the NID and looked for suspicious connectivity patterns.

161. The DIDS Director could respond to suspicious behavior by alerting a system security officer, who in turn could direct DIDS to actively respond to an attack including cutting off network access to a particular user. The DIDS Director could also direct the host and network monitors to modify their analysis via the "SET" command.

162. The NSM that formed the LAN monitor in DIDS was typically deployed at the gateway between the organization's internal hosts and the Internet. The largest deployment of network-based intrusion detection systems was by the Air Force Information Warfare Center (AFIWC) and NSM was deployed at an Air Force base's gateway between the base's internal networks and the Internet. This deployment in AFIWC was typically just in front of the firewall. Furthermore, *DIDS Oct. 1991* explicitly directs one to monitor at a gateway.

163. The DIDS papers described three levels of capability, each of which are represented in Figure 3:

- **DIDS Original Design** – DIDS, as described in *DIDS Oct. 1991*, was designed to support multiple LANs, each with its own set of LAN and host monitors. In addition to this being specifically mentioned in the text,[92] the LAN monitor reports to the Director included a field to identify *which* LAN monitor the report came from.

- **DIDS First Implementation** – While DIDS was designed to support multiple LANs, the first implementation to be delivered to the customer at the end of the first two-year contract was only tested on and guaranteed for a single LAN.

- **Planned Future DIDS** – *DIDS Oct. 1991* also described extending DIDS to include the addition of special-purpose monitors for network gateways and servers (i.e., file servers) and supporting additional levels of hierarchical monitoring.

164. Under Symantec's alternative claim construction of "monitor" the DIDS system is still anticipatory. The DIDS LAN monitor was software that could be reconfigured, for example, to add certain string matching capabilities. The DIDS Director was similarly software that could add such capabilities. The LAN monitors collected, analyzed and responded to suspicious network activity – using analysis engines and functionality to respond to the detection of events. The DIDS Director collected both suspicious events from the monitors, and low-level information (including network information) to track a user moving through a network. The DIDS Director correlated various suspicion reports and analyzed connectivity information for suspicious patterns. The DIDS Director GUI provided response by alerting the user of problems.

165. In my opinion, as more fully reflected in Exhibit D to my report, the DIDS system disclosed in *DIDS Oct. 1991 and DIDS Feb. 1991* satisfied every limitation of the indicated claims and enabled one of ordinary skill prior to November 9, 1997 to practice the inventions claimed in the patents-in-suit. Consequently, the noted claims of the patents-in-suit were not novel when filed.

---

[92] *DIDS Oct. 1991* at 169, *see also DIDS Feb. 1991* at 1 which refers to "LAN managers" indicating more than one network monitor, and 11 "each LAN segment has a LAN monitor…"

166. By 1991 DIDS was in public use. At trial I may testify regarding the use of DIDS on the UC Davis campus and at a government-owned location. In my opinion, as more fully reflected in Exhibit D to my Report, this public use of DIDS anticipates the noted claims of the patents-in-suit.

167. In addition, as reflect in Exhibit D, the remaining asserted claims of the patents-in-suit are invalid due to obviousness.

168. As described elsewhere in my report, it would have been obvious to one of ordinary skill in the art to combine DIDS with other intrusion detection projects—including NSM, GrIDS, and ISM—developed in whole or in part by researchers at UC Davis. Professor Karl Levitt, a highly-regarded researcher in intrusion detection, was the principal investigator for all of these projects and publicly disclosed these projects repeatedly in conferences and presentations. In fact, NSM served as the DIDS LAN Monitor, and DIDS was also part of the ISM architecture.[93] One of skill would have been motivated to combine these related systems, all of which were directed at detecting computer intrusions, to improve the functionality of the overall system.[94] See the NSM, ISM, and GrIDS charts for such references.

169. It would have also been obvious to one of ordinary skill in the art to use both the statistical profile-based anomaly detection of network traffic data practiced by DIDS and the signature-based detection of network traffic practiced by DIDS to analyze the network traffic data categories analyzed and described by: NetRanger, ISS RealSecure, Network Flight Recorder, Network Security Probe, Gabriel, synkill, and the SunScreen Firewall.[95] As shown in these references, all of these network traffic data categories were well-known, and it was also well-known that these categories could be

---

[93] *See ISM 1992.*
[94] *See* GrIDS webpage discussing using GrIDS with NSM. [SYM_P_0512095-2096].
[95] *See* the DIDS chart at Ex. D for the full titles and dates of all these references.

analyzed using either signatures detection, statistical profile-based anomaly detection, or both. One of skill would have been motivated to combine DIDS with these additional references in order to monitor additional types of network traffic for improved detection capabilities.

170. It would have been obvious to one of ordinary skill in the art to add to DIDS the response capabilities practiced or described by NetRanger, ISS RealSecure, CIDF, AIS, Network Security Probe, the '750 Patent, and synkill. Such response capabilities were well-known, and one of skill would have been motivated to combine DIDS with such response capabilities in order to expand the capabilities of DIDS to respond to intrusions.

171. It also would have been obvious to one of ordinary skill in the art to use DIDS to monitor a variety of different network entities, including virtual private network nodes and firewalls. Both *DIDS Feb. 1991* and *DIDS Oct. 1991* explicitly describe monitoring a variety of network nodes, and one of skill would have been motivated to expand the set of nodes monitored in order to increase the ability of DIDS to monitor all nodes in a network.

172. It would have also been obvious to one of ordinary skill in the art to extend the DIDS architecture to include peer-to-peer communications, as practiced by ISM, CSM, and Network Security Probe. Indeed, the DIDS was a part of the ISM architecture, and *ISM* 1992 described a peer-to-peer communications scheme. Furthermore, the CSM literature explicitly discusses DIDS.[96]

173. In addition, to the extent there were any differences between the configuration of DIDS and the configuration described in the patents-in-suit, it would

---

[96] *See* G. White et al., "*Cooperating Security Managers: A Peer-Based Intrusion Detection System*," IEEE Network, Jan./Feb. 1996.

have been obvious to modify the configuration of DIDS based upon the nature of the problem to be solved. Such configuration changes would have been motivated by a desire to address the problem of detecting and responding to suspicious network activity in very large enterprise networks.

174. Thus, as indicated in Ex. D, the remaining asserted claims of the patents-in-suit are obvious.

**3. ISM**

175. This report covers the Internet Security Monitor design, or ISM, as described in the paper "Internetwork Security Monitor: An Intrusion-Detection System for Large-Scale Networks," published in Proc. 15th National Computer Security Conference. Washington, DC, Oct. 1992, pp. 262-271 ("*ISM 1992*"). ISM had the following salient features:

- Built on NSM and DIDS capabilities and presumed familiarity with these systems (includes references).
- Correlated user's activity across domains through extension of DIDS' Network Identifier (NID).
- Developed the "thumbprint" technology to support NID construction with network-only monitors.
- Aggregated user's suspiciousness across domains.
- Aggregated host and service suspiciousness across domains.
- Supported peer-to-peer domain coordination.
- Supported hierarchical domain coordination.
- Balanced the value of aggregation with the need of privacy across domains.

176. UC Davis developed the ISM model as an extension to the Distributed Intrusion Detection System (DIDS). ISM's design was initially developed in 1992

towards the end of the first phase of DIDS, and it was conceived as a logical extension to the DIDS architecture. ISM first introduced the "thumbprint" concept (explained later in this section). UC Davis folded the NSM, DIDS and ISM concepts into their first DARPA-sponsored intrusion detection project, "Intrusion Detection for Very Large Networks," during which the thumbprint work was enhanced. A second paper on thumbprinting was published in 1995.[97] DARPA began funding the "Intrusion Detection for Very Large Networks" in approximately 1993. Eventually in the fall of 1995 the DARPA effort was redesigned into what became known as the Graph-based Intrusion Detection System (GrIDS), and work continued until about 1999.

177. *ISM 1992* explicitly states that ISM uses the DIDS and NSM systems and has an architecture "based on NSM and DIDs."[98] *ISM 1992* also points the reader to *DIDS Oct. 1991* and another paper on NSM: L.T. Heberlein, B. Mukherjee, K.N. Levitt, D. Mansur, "Towards Detecting Intrusions in a Networked Environment," Proc. 14th Department of Energy Computer Security Group Conference, pp. 17.47-17.65, May 1991 ("*NSM 1991*"). In addition, as noted previously, *DIDS Oct. 1991* explicitly directed the reader to *NSM 1990*. These references clearly state that NSM used both anomaly and misuse detection. Given this explicit direction to use *NSM 1990, NSM 1991* and *DIDS Oct. 1991*, these papers are incorporated into the text of *ISM 1992*. In the alternative, the combination of these references is obvious, as it is explicitly suggested by the references themselves.

178. DIDS was designed to detect and track attacks that spanned multiple users, multiple hosts, and multiple connections, but only within a single administrative domain. ISM was designed to detect and track attacks that spanned multiple domains. ISM

---

[97] *See* S. Staniford-Chen, and L.T. Heberlein *"Holding Intruders Accountable on the Internet"* Proc. of the 1995 IEEE Symposium on Security and Privacy, Oakland, CA, 8-10 May 1995, pp. 39-49.
[98] *ISM 1992* at 262, 263, 268.

presented extensions to the DIDS architecture to support the scalability and privacy concerns associated with coordinating multiple, independently administered domains.

179. *ISM 1992* introduced two extensions to support the ISM architecture: thumbprints for tracing users across unmonitored hosts and a set of protocols to extend "the Distributed Intrusion Detection System (DIDS) (see [Sna 91]) into arbitrarily wide networks."[99]

180. The thumbprint concept was designed to track users moving through any number of unmonitored hosts. In fact, two network connections can be compared even if there are an unknown number of intermediate hops separating the two connections being compared. For example, suppose an attacker begins at host H1, logs into host H2, from there logs into host H3, and so on until the attacker logs into host H10 from host H9. The thumbprint mechanism can compare connections H1->H2 and H9->H10 and determine they are related (i.e., it correlates them) even without any knowledge of all the connections from H2 to H9. Thus, the thumbprint approach extends DIDS' Network Identifiers (NIDs) across hosts without host monitors.

181. ISM's second extension to DIDS was a set of protocols to extend correlation and aggregation across multiple domains. The set of protocols could be divided into two groups: one to support peer-to-peer communications and another group to support hierarchical communications.

182. The peer-to-peer protocol extensions provided two capabilities: (1) to extend the tracking of a NID across multiple, independently administered domains and (2) to provide summaries of suspiciousness for NIDs, hosts, services, and vulnerabilities.

183. The hierarchical protocol extends the peer-to-peer communication by allowing hierarchically higher monitors to collect more information than peers can,

---

[99] *ISM 1992* at 264, citing to *DIDS Oct. 1991*.

namely, collecting additional details about a user's (NID) movements through the domain and lower-level events associated with that NID such as the number of files the user opened.

184. Figure 6 shows the basic approach. At the lowest level of the hierarchy, network monitors (1) and host monitors reported to a local ISM monitor such as the DIDS Director (2). These local ISM monitors could optionally report to ISM monitors in other administrative domains or to hierarchically higher ISM monitors (3). ISM monitors communicated with other ISM monitors in other administrative domains through peer-to-peer messaging (4) based on the CMIP protocol. ISM monitors communicated with hierarchically higher monitors (5) by extending the peer-to-peer messages with additional messages to collect details not available to peer monitors.

185. As the ISM had an "architecture based on NSM and DIDS," it used as data sources both network packets and host audit records; although with ISM's thumbprinting approach to track users, the ISM could support a network-only domain monitor.

186. At the lowest level of the ISM hierarchy, the host and network monitors responded by forwarding reports of suspicious reports to the local Director. For each subsequent level in the ISM hierarchy, or for peer-to-peer communications between ISM monitors, an ISM monitor responded by making the summary of their analyses available to other monitors.

187. *ISM 1992* presents an architecture that extends DIDS "into arbitrarily wide networks"[100] using both peer-to-peer and hierarchical communications. At the lowest level of the hierarchy, NSM-based monitors analyzed network traffic using a combination of anomaly and misuse detection. These network monitors reported to local ISM monitors. ISM monitors could communicate with other peer monitors in other

[100] *ISM 1992* at 264.

administrative domains or with hierarchically higher monitors. ISM monitors extended DIDS' tracking of uses via the NID concept across administrative domains, *and ISM 1992* proposed the concept of thumbprints to support this tracking across unmonitored hosts. ISM monitors could correlate and aggregate suspicious behavior based on NIDs, hosts, services, and vulnerabilities.

188. With regard to Symantec's alternative claim construction of "monitor," the prior analysis I provided on DIDS would apply to ISM as well. As noted in *ISM 1992*, at each level of the hierarchy there would be a security workbench. This security workbench would allow network managers to log in to their local ISM domain monitor and modify analyses. Furthermore, under SRI's definition of "statistical detection method" the ISM thumbprinting method would also qualify as a statistical detection method. However, under Symantec's claim construction for this term, thumbprinting would not qualify.

189. In my opinion, as more fully reflected in Exhibit E to my report, the ISM system disclosed in *ISM 1992* satisfied every limitation of the indicated claims and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the inventions claimed in the patents-in-suit. Consequently, the noted claims of the patents-in-suit were not novel when filed.

190. In addition, as reflected in Exhibit E, the remaining asserted claims of the patents-in-suit are invalid due to obviousness, for the same reasons as discussed previously regarding DIDS.

**4. GrIDS**

191. This section covers the Graph-based Intrusion Detection System (GrIDS system) and the prior art publications Staniford-Chen, S., et al. "GrIDS - A graph based intrusion detection system for large networks," 19th National Information Systems

Security Conference, 1996 ("GrIDS 1996") and "Steven Cheung, Rick Crawford, Mark Dilger, Jeremy Frank, Jim Hoagland, Karl Levitt, Stuart Staniford-Chen, Raymond Yip, Dan Zerkle, "The Design of GRIDS: A Graph-Based Intrusion Detection System," Technical report, UC Davis Department of Computer Science, Davis California (May 14, 1997) ("GrIDS 1997").

192. UC Davis conceived and developed the Graph-based Intrusion Detection System (GrIDS). UC Davis began development of GrIDS in 1995 as part of an initial 1993 DARPA contract to develop intrusion detection systems for very large networks. UC Davis continued the GrIDS work under a 1996 contract, and continued to work on it until about 1999. The GrIDS system was running publicly at UC Davis by approximately early 1997.

193. GrIDS had the following salient features:

- Correlated and aggregated intrusion detection reports via hierarchical collection of GrIDS monitors.
- Focused on detecting attacks that spanned multiple connections and across multiple administrative domains.
- Assumed the existence of lower-level sensors such as intrusion detection systems, but was not tied to any specific sensor type.
- Correlated reports from a single lower-level sensor (e.g., an IDS) or multiple lower-level sensors by building a graph of related reports.
- Aggregated reports from a single-lower level sensor (e.g., an IDS) or multiple lower-level sensors by summarizing attributes associated with the correlated reports.
- Reduced data amounts passed to hierarchically higher GrIDS monitors by only passing up relevant summarized information.
- Correlated reports from multiple lower-level GrIDS monitors arranged in a hierarchical fashion, typically organized to reflect an organization's hierarchical structure.

- Aggregated reports from multiple lower-level GrIDS monitors by integrating the summarized attributes from the lower-level GrIDS monitors into the graphs built by the higher-level GrIDS monitors.
- Detected intrusive activity by size and shape of graphs.
- Supported detecting intrusive activity by aggregating sensor-supplied attributes provided in individual reports.
- Configurable through built-in functions and operators to refine correlation and aggregation.
- Extensible through user-supplied functions to refine correlation and aggregation.
- Responded to suspicious behavior by sending messages to hierarchically higher monitors and possibly a user interface.
- Performed data reduction by sending only reduced graphs to hierarchically higher monitors.
- Described two network-based sensors for demonstrating GrIDS correlation and aggregation capability.

194. GrIDS was designed to detect and track attacks that spanned multiple domains. Whereas DIDS was designed to detect and track attacks that spanned multiple users, multiple hosts, and multiple connections, it only did so within a single administrative domain. Furthermore, the DIDS Director was tightly bound to the type of information generated by the lower-level sensor information. GrIDS, on the other hand, was largely designed to be independent of the lower-level sensors and could correlate and aggregate information from a wide variety of sources.

195. GrIDS was a hierarchical monitoring system for which the bottom-most layer of the hierarchy was made up of non-GrIDS sensors. These bottom-most sensors could be intrusion detection systems (for which there were a large number that had been developed and/or deployed by this point), simple network sniffers, or any other type of monitor that could send its output to a GrIDS monitor. GrIDS supported a well-defined and public protocol, including the "GrIDS Common Packet Format (GCPF)" and a well-

defined syntax, so third-party sensors could be extended to support GrIDS or the sensors could be "wrapped" with a second program that translated the sensor's proprietary message into a GrIDS message.

196. Once a GrIDS monitor received a report from a lower level sensor, it attempted to correlate the new report with previously received reports. If a correlation could be found, the new report was aggregated with the previous reports. GrIDS performed correlation and aggregation through the concept of a graph – a computer science term referring to a data structure consisting of nodes and edges connecting nodes. Each new report was turned into a graph; for example, a report of a network attack from host A to host B would be turned into a report with nodes representing hosts A and B and an edge between these nodes representing the communication between these nodes (the edge could be "annotated" with the specific type of attack). This new graph representing the report was compared to existing graphs representing past reports, and if the new graph shared a node or edge with an existing graph, the new graph was "correlated" with this existing graph, and the existing graph was extended by "aggregating" or "integrating" the new graph with it.

197. GrIDS monitors used a number of heuristic and potentially user-supplied functions to determine if a given graph was suspicious and should be forwarded to higher-level GrIDS monitors or user interfaces. An example heuristic was a graph that exceeded a particular size in the number of nodes or number of edges. For example, if the graph grew too large, it indicated a potential worm or attacker moving through the network, and that graph would be determined to be suspicious. If a graph took on a particular shape, that would indicate a sweep or other pattern of misuse, and again the graph would be flagged as suspicious.

198. If a GrIDS monitor determined that a graph had become suspicious, the GrIDS monitor automatically responded by forwarding a reduced version of the graph to

a hierarchically higher GrIDS monitor. It could also send an alert to a graphical user interface or any other subject that subscribed to receive alerts.

199. Hierarchically higher GrIDS monitors applied the same algorithms and code, but they applied them to the graphs passed up from hierarchically lower GrIDS monitors (as opposed to sensors such as traditional intrusion detection systems).

200. Figure 7, based on *GrIDS 1997* figures 1.3 and 1.4, demonstrates several features of GrIDS. Fig. 7 shows the College of Engineering at a university, and the College is composed of three departments – Electrical Engineering (EE), Computer Science (CS), and Civil Engineering (CE). Each department has its own GrIDS monitor receiving messages from one or more sensors. Each department GrIDS monitor sends alerts (consisting of reduced graphs) to the hierarchically higher college of engineering GrIDS monitor.

201. In Fig. 7, the hosts are labeled 'A' through 'J'. Host 'A' in Civil Engineering initiates a connection to host 'B' (1). Shortly after this activity, host 'B' initiates a connection to host 'H' in Electrical Engineering (2). Host 'H' then initiates connections to hosts 'J' (3) and 'I'. The sensor (4) in EE detects these last several connections and based upon its own analysis reports them as suspicious via standard GrIDS messages (5) to the department's GrIDS monitor (6). The EE GrIDS monitor builds the graph shown in (7). It determines this graph is suspicious and sends a GrIDS message (8) to the hierarchically higher college GrIDS monitor (9). The college monitor builds its own graph (10).

202. Fig. 7 demonstrates a 3-tier hierarchy consisting of the lowest level sensor (4), the department GrIDS monitor (6), and the college GrIDS monitor (9). It also demonstrates correlation and aggregation through the construction of graphs from simpler reports (7) and (10). It also demonstrates communication over well-defined and open

protocols (5) and (8). It also demonstrates data reduction as the hierarchically higher college GrIDS monitor builds graphs from reduced data graphs (10).

203. GrIDS monitors processed messages encoded in GrIDS Common Packet Format (GCPF), and the data messages were either generated by lower-level sensors or hierarchically lower GrIDS monitors. The initial messages could "come from other IDSs, network sniffers, or any monitor that is equipped with a filter to send its output to GrIDS."[101]

204. *GrIDS 1997* describes a prototype network sniffer to demonstrate the GrIDS approach.[102] The sniffer analyzed TCP connections and UDP sessions. For TCP connections and UDP sessions the sniffer generated event reports for:

- Start of TCP connection
- Normal close of connection via FIN flags
- Close of connection via RST flags
- Close of connection because of a timeout
- Start of UDP sessions
- End of UDP sessions
- ICMP error messages

205. Furthermore, for Telnet connections the sniffer reported several additional Telnet protocol specific events including:

- START
- OPTION NEGOTIATION
- AUTHENTICATION

---

[101] *GrIDS 1997* at 7.
[102] *GrIDS 1997* at Chapter 7.

- DATA
- END
- RESET

206. Finally, for NFS sessions the sniffer reported several NFS protocol specific events including:

- NFS Authentication error
- NFS Stale file handle error
- NFS mount success
- NFS mount failure
- Set UID error
- Read success
- Write success

207. The proposed sniffer analyzed network packets, identified and tracked individual sessions (connections), and detected and reported the above-mention events. Each GrIDS monitor mapped a report from a lower-level sensor or GrIDS monitor into a graph and correlated and aggregated the new graph with the previous graphs as described above. A GrIDS monitor could measure several features of a graph to determine if it was suspicious (e.g., graph size) or use user-supplied functions to analyze the graph.

208. With regard to Symantec's alternative claim construction of "network monitor," the GrIDS graph engine would satisfy all the requirements for such a monitor. GrIDS graph engines could be dynamically reconfigured: for example, they could add hosts, add departments, or even accept an entire new rulebase. GrIDS graph engines collected, analyzed and responded to suspicious network activity. GrIDS graph engines included an analysis engine and a mechanism for response. GrIDS graph engines would

in fact use the same code base for all of the engines at each level of the hierarchy, and thus would also satisfy ISS's claim construction requirement of "generic code."

209. In my opinion, as more fully reflected in Exhibit E to my report, the GrIDS system disclosed in *GrIDS 1996* and *GrIDS 1997* satisfied every limitation of the indicated claims and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the inventions claimed in the patents-in-suit. Consequently, the noted claims of the patents-in-suit were not novel when filed.

210. Furthermore, GrIDS—as well as NSM and DIDS—was in public use prior to November 9, 1997. In my opinion, as more fully reflected in Exhibit E, this public use anticipates the indicated claims from the patents-in-suit.

211. In addition, as reflect in Exhibit E, the remaining listed claims of the patents-in-suit are invalid due to obviousness. As described elsewhere in my report, it would have been obvious to one of ordinary skill in the art to combine GrIDS with other intrusion detection projects—including NSM, DIDS, and ISM—developed in whole or in part by researchers at UC Davis.

212. It also would have been obvious to one of ordinary skill in the art to use GrIDS to monitor a variety of different network entities—including routers, gateways, and firewalls—as practiced or described by DIDS, the SunScreen Firewall, and NetRanger. A critical function of the GrIDS sensor is to monitor traffic in and out of its domain, and the best place to do that is at the gateway.

213. In addition, to the extent there were any differences between the configuration of GrIDS and the configuration described in the patents-in-suit, it would have been obvious to modify the configuration of GrIDS based upon the nature of the problem to be solved. Such configuration changes would have been motivated by a

desire to address the problem of detecting and responding to suspicious network activity in very large enterprise networks.

214. Thus, as indicated in Exhibit E, the indicated asserted claims of the patents-in-suit are obvious.

**5. JiNao**

215. This section covers the JiNao design as described in the technical report by F. Jou, "Architecture Design of a Scalable Intrusion Detection System for the Emerging Network Infrastructure," dated April 1997 ("*JiNao Report*"), as well as the associated slides indicated on Exhibit G. MCNC and North Carolina State University (NCSU) performed this work. The JiNao design provided a scalable architecture to detect and protect against both known and unknown attacks against the network infrastructure.

216. The architecture described, collectively called JiNao, used a combination of anomaly detection, misuse detection, correlation, and aggregation to detect intrusive activity. The lowest level JiNao monitors (described as a "Local JiNao") analyzed packets, while the higher level monitors (described as "Remote Management Applications") analyzed the results of Local JiNao monitors or other higher level monitors. Communications between monitors was via the standard Simple Network Management Protocol (SNMP) and Management Information Base (MIB) architecture. JiNao also uses the standard SNMP network protocol and MIB architecture to support hierarchical and peer-to-peer communications to provide scalability and detect larger scale attacks. The JiNao architecture was independent of any particular network protocol (the modules use the generic term "Protocol Data Unit" (PDU)), but the developers planned to deliver implementations for analyzing the OSPF routing protocol and the

SNMP management protocol.[103] By placing Local JiNao monitors on routers or next to routers, JiNao could respond to attacks by blocking offending packets. Furthermore, the authors planned to track the DARPA joint effort called Common Intrusion Detection Framework (CIDF) to support interoperability and module reusability with other DARPA sponsored projects.[104]

217. Thus, JiNao had the following salient features:

- Processed network packets to look for intrusive behavior and network faults.
- Ran on or next to routers and gateways.
- Designed to be independent of any particular network protocol.
- Initial implementation would analyze the OSPF routing protocol and the SNMP management protocol.
- Analyzed network traffic with both misuse and anomaly detection engines.
- Used NIDES statistical algorithms for anomaly detection.
- Correlated the results of the misuse and anomaly detectors.
- Correlated encrypted and decrypted versions of the same packet to enhance analysis.
- Responded to attacks by alerting higher-level monitors (via SNMP trap messages), modifying analysis, and blocking traffic.
- Modified analysis based on events detected by other sensors and monitors.
- Exchanged information with other monitors using a standardized network protocol (SNMP) and databases (MIB).
- Used the same misuse and anomaly detection approaches at hierarchically higher monitors.

---

[103] *JiNao Report* at 5-6, 14, 15.
[104] *JiNao Report* at 13-14.

- Supported peer-to-peer and hierarchical monitoring for higher-level monitors.
- Correlated and aggregated/integrated information at regional and global levels.
- Designed to integrate with additional network management or security applications by using the standards-based SNMP framework.
- Designed to support interoperability and reusability with other DARPA-sponsored projects efforts via the emerging Common Intrusion Detection Framework (CIDF).

218. Figure 8 shows a composite view of the fully scalable architecture of JiNao as described in the *JiNao Report*. It shows several "Local JiNao" systems at the bottom that monitor individual network routers and several higher-level monitors communicating in a peer-to-peer and hierarchical architecture. The MCNC/NCSU contract with DARPA for which this report was written only required them to implement the local JiNao components.[105]

219. In the lower left portion of Figure 8 is a detailed description of a local JiNao system for a single router. At the bottom of this local JiNao system is the Intercept/Redirect module (1) which intercepted a network packet before the router's software could process it, forwarded the packet to the analysis modules (2) (3) (4), and if (2) did not complain, would forward the packet to the router's software. If JiNao determined that a packet was part of an attack, this module could respond by preventing the malicious packets from being sent to the router's software.

220. The Intercept/Redirect module forwarded the packet to the next module in the JiNao system, the Simple Misuse Detection module (2). The *JiNao Report* actually calls this the rule-based "prevention module," and it applied simple rules to a single packet in order to quickly detect clear security violations.

---

[105] *JiNao Report* at 3.

221. The Simple Misuse Detection module (2) forwarded the packets to the local detection module that consisted of an anomaly detector (3) and a misuse detector (4) that detected attacks that spanned multiple packets. The anomaly and misuse detectors forwarded their results to the Local Decision Module (5), which correlated information from the other sensors and determined whether a response should be taken. Potential responses included directing the Simple Misuse Detector (5) to start blocking certain packets, adjusting the analysis of the misuse and anomaly detectors, making the analysis available to other monitors through a MIB (6), and alerting high-level monitors through an SNMP trap message.

222. A local JiNao system communicated with a hierarchically higher monitor through the network agent (7) using a standardized SNMP protocol (8). JiNao's local results and configuration information were available to higher-level monitors through a Management Information Base (MIB) (6), and alerts were automatically sent to higher-level monitors though SNMP trap messages, part of the standardized SNMP network management architecture.

223. The JiNao architecture supported hierarchically higher monitors called Remote Management Applications (9) that used the same anomaly and misuse algorithms (10) as the local JiNao monitors. Remote Management Application monitors could communicate peer-to-peer with other Remote Management Application monitors (11) via the standard ManagerToManager portion of the SNMP protocol to aggregate results to support regional detection. Furthermore, Remote Management Application monitors could communicate hierarchically (12) to develop a more global view of potential misuse in the network. Although the *JiNao Report* states that a hierarchical system had not yet been created, the report is clear that the architecture described was suitable for implementing such a hierarchy. Furthermore, because the *JiNao Report* explicitly directed one of skill to use the known hierarchical architecture of the SNMP framework,

one of skill in the art would have understood this reference to disclose and enable hierarchical JiNao monitors.

224. JiNao processed network packets. The messaging infrastructure for internal JiNao modules used a generic Protocol Data Unit (PDU), so they could be tailored for any protocols. MCNC/NCSU proposed to deliver solutions for the OSPF and SNMP protocols, and the *JiNao Report* mentions the possibility of using JiNao to process HTTP packets to protect web servers. In addition, the *JiNao Report* explicitly states that JiNao monitored data volume.[106]

225. JiNao analyzed network traffic through a combination of anomaly detection based on the NIDES statistical algorithms and misuse detection. The Simple Misuse Detection module (labeled (2) in Figure 8 and called "Prevention Module" in the report) used simple rules that could quickly evaluate individual packets for clear security violations. The Stateful Misuse Detection Module (labeled (4) in Figure 8 and called "Protocol Analysis" in the report) used a collection of Finite State Machines (FSMs) to detect more complex attacks that can span multiple packets. The anomaly detector (3) was based on the NIDES statistical algorithms.

226. The Protocol Analysis module was a collection of finite state machines (FSM). Each FSM was a signature designed to detect a specific attack. UC Davis had also done research on methods to detect attacks through protocol analysis, and thus I am familiar with this type of analysis. Typically, such analysis sets forth a set of rules describing the "correct" behavior of a particular protocol, and then looks for deviations from this known proper behavior. Such analysis may also look for known exploitations of known vulnerabilities of a particular protocol.

---

[106] *JiNao Report* at 19.

227. The Local Decision Module (labeled (5) in Figure 8) correlated the results from the anomaly and misuse detectors. Hierarchically higher monitors would use similar detection/analysis functions to "correlate intrusion events among several routers."[107] For example, as described in the *JiNao Report*, JiNao could correlate conflicting information from multiple JiNao systems to determine that a router between those two systems (Bob) was bad.[108]

228. JiNao could respond to suspicious events in several ways. First, the Local Decision Module (5) could update the MIB Database (6), thus making the information available to hierarchically higher monitors. Second, the Local Decision Module could immediately automatically alert hierarchically higher monitors with an SNMP trap message. Third, the Local Decision Module could direct the local analysis modules to modify their analyses. Fourth, the Local Decision Module could raise an alert to the security officer via an email or a graphical user interface. Fifth, the Local Decision Module could interact directly with the router to modify its settings. Sixth, the Simple Misuse Detection module (labeled (4) in Figure 8 and called "Prevention Module" in the document) could direct the Interception/Reception module (1) not to forward the packet to the router's software.

229. The JiNao modules would satisfy the "monitor" limitation under any of the parties' definitions of that term.

230. In my opinion, as more fully reflected in Exhibit G to my report, the JiNao system disclosed in the *JiNao Report* and related slides satisfied every limitation of the asserted claims of the patents-in-suit (except for '338 claims 2-3, 5-8 and 10) and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the alleged

---

[107] *JiNao Report* at 13.
[108] *JiNao Report* at 35.

inventions claimed in the patents-in-suit. Consequently, the noted claims of the patents were not novel when filed.

231. In addition, as reflect in Exhibit G, all of the remaining asserted claims of the patents-in-suit are invalid due to obviousness.

232. It would have been obvious to one of ordinary skill in the art to use JiNao's statistical profile-based anomaly detection of network packets to analyze the additional network traffic data categories analyzed and described by: "A Network Security Monitor—Final Report," NetRanger, ISS RealSecure, Network Flight Recorder, Network Security Probe, Gabriel, synkill, SNMP/RMON IETF standards, and the SunScreen Firewall.[109] This is particularly true since the JiNao Report explicitly mentions monitoring SNMP.[110] As shown in these references, all of these network traffic data categories were well-known, and it was also well-known that these categories could be analyzed using either signature detection, statistical profile-based anomaly detection, or both. One of skill would have been motivated to combine JiNao with these additional references in order to monitor additional types of network traffic for improved detection capabilities.

233. In addition, to the extent there were any differences between the configuration of JiNao and the configuration described in the patents-in-suit, it would have been obvious to modify the configuration of JiNao based upon the nature of the problem to be solved. Such configuration changes would have been motivated by a desire to address the problem of detecting and responding to suspicious network activity in very large enterprise networks.

---

[109] *See* the NSM chart at Ex. C for the full titles and dates of all these references.
[110] *JiNao Report* at 5-6, 15, 17, 18

### 6. EMERALD 1997

234. This section covers the EMERALD design as described in the paper titled "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances," published in the Proceedings of the 20th National Information Systems Security Conference in October of 1997 ("*Emerald 1997*"). Phil Porras, one of the named inventors of the patents-in-suit, and Peter Neumann are the authors of this paper. The primary sponsor for this work was the Department of Defense via the Defense Advanced Research Projects Agency (DARPA).

235. *Emerald 1997* describes an architecture for monitoring an enterprise network collectively called EMERALD. EMERALD analyzed an input stream such as network packets, and applied anomaly detection, misuse detection, correlation, and aggregation to detect possible misuse. EMERALD's internal modules used a well-defined API to pass messages between modules; EMERALD also used a well-defined protocol to support hierarchical and peer-to-peer communications to provide scalability and detect larger scale attacks.

236. EMERALD as described in this publication had the following salient features:

- Monitored network services and network components such as routers and gateways.
- Analyzed network packets, audit logs, application logs, and results of other intrusion detection sensors; these data sources are referred to as an event stream.
- Used anomaly detection based on the NIDES algorithms to analyze the event stream.
- Used signature detection to analyze the event stream for known examples of misuse.
- Provided an Application Programming Interface (API) to allow third-party modules and sensors to participate in an EMERALD system.

- Correlated analysis from the anomaly detection engine, the signature detection engine, and potentially other analysis engines.
- Supported hierarchical monitoring beginning at the lowest level (service monitor), with the next higher level being a domain monitor, and the highest level being an enterprise monitor.

237. The system described in *Emerald 1997* shared the basic elements of SRI's previous work in IDES and NIDES, but it was designed to be distributed and independent of any particular data source. The paper states that EMERALD's distributed design addresses scalability. Designing the system to be independent of the underlying data source makes it easier to incorporate other data sources, including results from other intrusion detection sensors.

238. EMERALD took the basic NIDES architecture – an anomaly engine, a signature/misuse engine, and a resolver – and generalized it for any input stream. The earlier SRI intrusion detection systems (IDES, NIDES and Safeguard) shared a similar architecture, but they were largely focused on an input stream of host-generated data (e.g., audit trails) to analyze host-related subjects (e.g., user accounts and processes). EMERALD generalized the approach so that the system was not tied to any specific input stream or subject matter.[111]

239. Furthermore, because the input stream was generalized, the output event stream for one monitor could be used as the input stream for another monitor. Because EMERALD monitors could be composed in this manner, a hierarchy of monitors could be created, and this hierarchical approach in turn created the ability to scale the analysis to larger environment.

240. Figure 9 shows a composite view of the fully scalable architecture of the EMERALD system as described in *Emerald 1997*.

---

[111] *Emerald 1997* at Fig. 1 shows the "generic EMERALD monitor architecture."

241. In the lower left portion of Figure 9 is a description of an EMERALD "Service Monitor" (1). It read input streams such as network packets (2), and then analyzed that input stream with an anomaly detection engine (3) and a signature detection engine (4). The results of these analyses engines were passed to the Resolver (5), which could correlate the results, take some type of response (6), or pass its analysis to hierarchically higher monitors (7). The boxes to the right of the annotated Service Monitor (1) are simply other Service Monitors.

242. The next level of monitor in the hierarchy was called the "Domain Monitor" (8), and it analyzed and correlated the results (7) from the lower level Service Monitors (1). The Domain Monitor shared the same architecture as the Service Monitor, including an anomaly detection engine, a signature/misuse detection engine, and a Resolver. The Domain Monitor could also pass the results of its analysis to other Domain Monitors (9) via peer-to-peer sharing, or it could pass the results to a hierarchically higher Enterprise Monitor (10).

243. The top level of monitor in the hierarchy was called the "Enterprise Monitor" (11). The Enterprise Monitor analyzed and correlated the Domain Monitors' results (10). The Enterprise Monitor shared the same architecture as the Domain and Service Monitors, namely, an anomaly detection engine, a signature/misuse detection engine, and a resolver.

244. EMERALD processed any type of input stream. Examples in the *Emerald 1997* paper included audit data, network datagrams (i.e., packets), SNMP traffic, application logs, and results from other intrusion detection sensors.[112]

245. EMERALD used a combination of anomaly detection, signature detection, and a resolver to analyze the input streams. The signature/misuse engine detected known

[112] *Emerald 1997* at 356.

intrusive behavior. The anomaly engine detected unusual activity that might indicate previously unknown intrusive behavior. The resolver integrated and correlated information from the anomaly and signature engines and potentially from other monitors as well.

246. The signature analysis engine is described as mapping an incoming event stream against "abstract representations of event sequences that are known to indicate undesirable activity."[113] The objectives of the signature analysis depend on which level in the hierarchy the analysis is operating at: service monitors target known attacks on network services and infrastructure, whereas above the service layer, the signature engines scan aggregated reports for more global coordinated attack scenarios.

247. The anomaly detection engines are described as using the statistical profiling type of anomaly detection developed in NIDES. The paper notes that the "underlying mechanisms" of NIDES are "well suited to the problem of network anomaly detection, with some adaptation."[114] *Emerald 1997* then described the modifications to the statistical profiling mechanism needed in order to achieve the generality of input desired in EMERALD.

248. EMERALD's resolver was responsible for responding to suspicious behavior. The paper describes several different actions the resolver could take in response to detected suspicious activity including: (1) closing connections, (2) terminating processes, (3) calling a host's integrity checker to verify operating state, (4) propagating information to other monitors, (5) modifying the analysis of its detection engines, and (6) sending information to the user interface.[115]

---

[113] *Emerald 1997* at 359.
[114] *Emerald 1997* at 359.
[115] *Emerald 1997* at 361.

249. The common specification of the patents-in-suit is remarkably similar to the text of the *Emerald 1997* paper. In order to assist my comparison of these two documents, I have reviewed a highlighted comparison of the two documents (see Ex. GG).[116] My review of this comparison indicates that a great deal of the text of *Emerald 1997* was reproduced verbatim or nearly verbatim in the specification of the patents-in-suit. In addition, Figures 1 and 2 of *Emerald 1997* are virtually identical to Figures 2 and 3 of the patents. I understand that one of the inventors, Mr. Porras, described *Emerald 1997* as merely an "early conceptual design."[117] This position is not plausible in light of the substantial overlap between the patents and this paper. I have examined the portions of the specification not present in the *Emerald 1997* paper and I do not believe that they provide any important disclosure required to implement the system described beyond what *Emerald 1997* disclosed in light of the knowledge of one of ordinary skill in the art.

250. Furthermore, given the similarity in language between *Emerald 1997* and the patents, my opinions on invalidity and obvious with regard to *Emerald 1997* and all other NIDEs and Emerald –related references do not change regardless of which claim construction position is ultimately adopted by the Court.

251. SRI has claimed that *Emerald 1997* does not teach the specific categories of network traffic data called out in, for example, '338 claim 1, '203 claim 1, and '615 claim 1.[118] First, it is important to point out that monitoring these particular types of "measures" of network packets or "network traffic data" was not new or novel – all of

---

[116] This comparison chart was prepared for demonstrative purposes only, and is not intended to be a strict word-for-word comparison. Minor differences in highlighted language may exist.
[117] Porras Tr. 433.
[118] *See* SRI International, Inc.'s "Amended" Response to Symantec's Invalidity and Inequitable Conduct Contentions.

them had been monitored by other systems before. The inventors have acknowledged this fact for many of the claimed categories.[119]

252. Second, as explained in the Expert Report of Frederick Avolio, *Emerald 1997* explicitly stated that the disclosed EMERALD system could monitor and analyze "network infrastructure" including firewalls. *Emerald 1997* also disclosed monitoring an event stream from an application log, which would include a firewall log. Firewalls in 1997 provided a common set of monitoring and logging features which were well-known in the art at the time. Thus, one of skill would have understood from the *Emerald 1997* disclosure that one should monitor network connections, including network connection requests and denials, and data transfers, including network packet data volume/network packet data transfer volume. I have reviewed Mr. Avolio's report and spoken with him in detail about it. I agree with and adopt in my own report his report, including his analysis and conclusions. I also adopt the references he relied upon in reaching his conclusions.

253. In addition, *Emerald 1997* disclosed monitoring an event stream of SNMP traffic[120] and monitoring network infrastructure.[121] As explained more fully in my discussion of HP Openview and the Internet standards (RFCs), SNMP traffic and related network infrastructure management data provides monitoring for a variety of different categories of network traffic data. The claimed categories of network traffic data would have been understood by one of ordinary skill in the art to be disclosed by this explicit reference to monitoring SNMP traffic and network infrastructure. See also my table at Exhibit X.

254. Furthermore, the types of network traffic one should monitor to detect suspicious activity or network intrusions flow naturally from the types of attacks that one

---

[119] Porras Tr. 289-295, 444-454; Valdes Tr. 283-287.
[120] *Emerald 1997* at 356.
[121] *Emerald 1997* at 355.

is looking for. As the intrusion detection and computer security fields developed, practitioners in the field began cataloging and tracking security-related intrusions seen by different computer networks. For example, after the Morris worm incident in November 1988, which caused extensive damage to different internet systems, DARPA called on the Software Engineering Institute at CMU to set up a center to coordinate communication among experts during security emergencies.[122] Known as the Computer Emergency Readiness Team ("CERT"), CERT issued public advisories to warn of new attacks or vulnerabilities.

255. For example, in September 1996, CERT first issued a warning about TCP SYN flooding and IP Spoofing attacks.[123] This advisory explained the SYN flooding attack, in which numerous requests to open a TCP connection are sent that cannot be responded to, flooding the system with half-open connections and making it difficult for the system to continue to communicate. One of skill in the art at the time would have known about SYN flooding attacks and understood that one should monitor network connection requests and denials to detect such an attack. A variety of other types of network attacks were also well-known at the time, which would have similarly indicated different categories of network traffic to monitor. [124]

256. To the extent that the various categories of network traffic data or measures of network packets are not disclosed in *Emerald 1997*, it would have been obvious to one of skill in the art to combine *Emerald 1997* with any of the many well-known firewalls, intrusion detection research systems, commercial intrusion detection

---

[122] http://www.cert.org/meet_cert/meetcertcc.html
[123] CERT Advisory CA-1996-21 [SYM_P_0548726-734]. *See also* Schuba et al., "Analysis of a Denial of Service Attack on TCP," Proc. of the 1997 IEEE Symposium on Security and Privacy, Oakland, CA, 208-23 (May 4-7 1997) [SYM_P_0535408-28].
[124] Numerous other attacks were well-known at the time. *See, e.g.*, CERT Advisory CA-1996-26 "Denial-of-Service Attack via ping, Dec. 18, 1996 [SYM_P_0548718-725] (discussing ICMP error packets and oversized ICMP datagrams).

systems, and IETF defined standards on what to monitor for network infrastructure, all of which were already monitoring these network traffic data categories. In fact, *Emerald 1997* explicitly pointed the reader to other systems, including NSM.[125] For example, both ISS RealSecure and NetRanger were well-known network intrusion detection systems, and one of skill upon reading *Emerald 1997* would have been motivated to look at the traffic data being monitored by such systems in order to select the best categories of traffic for use with EMERALD. Indeed, I understand that the inventors in 1997 were actually working with and looking at Sun Microsystem's SunScreen firewall, further supporting my opinion that firewalls and other IDS systems would have been obvious sources of information about useful network traffic data categories.[126] Since the SunScreen firewall and NetRanger also included the use of VPNs, it similarly would have been inherent or obvious to combine the system of *Emerald 1997* with the VPNs in use by these products.

257. I also understand that SRI claims that the *Emerald 1997* paper does not provide an "enabling disclosure" of a statistical detection method.[127] I disagree. *Emerald 1997* included an entire section entitled "Scalable Profile-Based Anomaly Detection" which describes how EMERALD will incorporate and modify the well-known NIDES algorithms for statistical profiling. *Emerald 1997* also pointed the reader to additional references on the NIDES algorithms. Furthermore, other additional SRI publications had also already disclosed the NIDES algorithms in detail.[128] To the extent that the specific

---

[125] *Emerald 1997* at 364.
[126] Valdes Tr. 68-69; 12-123. *See also Emerald – Conceptual Design 1997* at p. 88. [SRI 012400].
[127] *See* SRI International, Inc.'s "Amended" Response to Symantec's Invalidity and Inequitable Conduct Contentions.
[128] *See, e.g.*, A. Valdes and D. Anderson, *Statistical Methods for Computer Usage Anomaly Detection Using NIDES*, Proc. of the Third International Workshop on Rough Sets and Soft Computing, January 1995 ("*Statistical Methods*") [SYM_P_0068937-942]. Mr. Valdes testified that the equations disclosed in this paper were suitable for network monitoring. Valdes Tr. 344-346. Furthermore, the patents' specification explicitly states

algorithms for statistical profiling are not incorporated-by-reference into *Emerald 1997*, it would have been obvious to one of skill in the art to combine *Emerald 1997* with prior NIDES-related publications, since *Emerald 1997* states that the NIDES algorithms form the basis for the statistical profiling disclosed. One of skill would have been motivated to further investigate these specific algorithms in order to improve the performance of the system.

258. In my opinion, as more fully reflected in Exhibit H to my report, the system disclosed in *Emerald 1997* satisfied every limitation of each of the asserted claims of the patents-in-suit (except '338 claim 7) and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the inventions claimed in the patents-in-suit. Consequently, the noted claims of the patents-in-suit were not novel when filed.

259. In addition, as reflected in Exhibit H, the remaining indicated claim is invalid as obvious, for the reasons detailed above.

**a. CMAD and Conceptual Overview**

260. In addition to the *Emerald 1997* paper, SRI also published two shorter earlier descriptions of the EMERALD system. *See* P. Neumann, P. Porras and A. Valdes, "Analysis and Response for Intrusion Detection in Large Networks," Summary for CMAD Workshop, Monterey, 12-14 November 1996 ("Emerald CMAD") and P. Porras and P. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances Conceptual Overview," December 18, 1996 ("Emerald – Conceptual Overview"). While neither of these references is as detailed as *Emerald 1997*, both also anticipate or render obvious many of the asserted claims of the patents-in-suit for the reasons stated above.

---

that the techniques described in this paper may be used for the "profile engine" which "can profile network activity via one or more variables called measures." '338 col. 5:43-50.

**b. Live Traffic Analysis and Conceptual Design 1997**

261. SRI also published two additional papers regarding the EMERALD system: P. Porras and A. Valdes, "Live Traffic Analysis of TCP/IP Gateways," http://www.sdl.sri.com/projects/emerald/live-traffic.html, Internet Society's Networks and Distributed Systems Security Symposium, Nov. 10, 1997 ("*Live Traffic Analysis*") and P. Porras and P. Neumann, "Conceptual Design and Planning for EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances," Version 1.2 20 May 1997 ("Emerald - Conceptual Design 1997"). I understand that there is a debate over whether or not these documents were publicly available prior to November 9, 1997. If public, *Live Traffic Analysis* anticipates all of the asserted claims of all of the patents-in-suit, except for '338 claim 25, which is rendered obvious, as indicated in Exhibit L. If public, Emerald – Conceptual Design 1997 anticipates or renders obvious all of the asserted claims of the patents-in-suit, for the reasons discussed above and those noted on Exhibit K.

**c. Emerald 1997, Intrusive Activity 1991, NIDES 1994**

262. *Emerald 1997* also references two additional publications in both the text of the paper itself, and in the list of references. *Emerald 1997* explains that the statistical algorithms in H. Javitz and A. Valdes, "The NIDES statistical component description and justification," Technical report, Computer Science Laboratory, SRI International, Menlo Park, CA March 1994 ("*NIDES 1994*") provides the foundation for the profile-based anomaly detection in *Emerald 1997*:

> "Requirements for an anomaly-detection system that became IDES were documented in [6]. This research led to the development of the NIDES statistical profile-based anomaly-detection subsystem (NIDES/Stats), which employed a

> wide range of multivariate statistical measures to profile the behavior of individual users [9]."[129]

263. In addition, *Emerald 1997* also directs one to the Network Security Monitor (NSM) system for analysis of network traffic, and identifies L.T. Heberlein, B. Mukherjee, K.N. Levitt., "A Method to Detect Intrusive Activity in a Networked Environment," Proc. 14th National Computer Security Conference, pp. 362-371, Oct. 1991 ("*Intrusive Activity 1991*") as describing NSM:

> "[T]he Network Security Monitor [7] seeks to analyze packet data rather than conventional audit trails…"[130]

264. Given these explicit references in *Emerald 1997* to both *NIDES 1994* and *Intrusive Activity 1991*, these three references should be considered to be a single disclosure for purposes of anticipation. In the alternative, the citations above provide a motivation to combine these three references in order to make and improve the statistical profiles and network traffic monitoring claimed in the patents-in-suit. As shown in Exhibit U, *Emerald 1997*, *NIDES 1994* and *Intrusive Activity 1991* together satisfied every limitation of the asserted claims of the patents-in-suit (except claim 7, which is invalid as obvious) and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the alleged inventions claimed in the patents. Consequently, the noted claims of the patents were not novel when filed.

### 7. Network NIDES

265. This section covers NIDES and future design work (collectively called "Network NIDES" in this section) as described in the report titled "Next-generation Intrusion Detection Expert System (NIDES) A Summary," dated May of 1995 ("*Network NIDES*"). SRI International performed this work, with Debra Anderson, Thane Frivold, and Alfonso Valdes (one of the named inventors) listed as the authors of this report. The

---

[129] *See Emerald 1997* at 359.
[130] *See Emerald 1997* at 364.

primary sponsor for this work was the Department of the Navy, Space and Naval Warfare Systems Command (SPAWAR).

266. NIDES was under development from about 1992 through late 1994, with this report delivered in 1995, although NIDES itself was essentially an extension to SRI's Intrusion Detection Expert System (IDES), which dates back to the mid-late 1980s. NIDES was designed to detect intrusive behavior using a combination of anomaly and misuse detection. NIDES was essentially a redesign of the SRI's earlier IDES, with a focus on a modular and more flexible approach. This modularization and flexibility also helped retarget NIDES to other platforms than its original Solaris audit-based analysis. For example, NIDES was modified to support (1) monitoring of Trusted Xenix systems (Advisor Project), (2) database transaction logs (FBI FOIMS-IDES Project), (3) and analyzing applications (Safeguard Project).[131]

267. The basic NIDES architecture included an anomaly engine performing statistical profiling, a signature/misuse engine, and a resolver. The anomaly detection engine detected previously unknown intrusive activity based on the assumption that this behavior would appear different than normal behavior. The signature/misuse detection engine detected known intrusive activity. The resolver received the analysis from the anomaly and misuse detection engines, integrated multiple reports associated with the same underlying event, and responded by sending a message to the user interface or sending out email.[132]

268. Network NIDES had the following salient features:

- Monitored network components such as routers and gateways.

[131] *Network NIDES* at 3.
[132] *Network NIDES* at 22-23.

- Analyzed network packets, audit logs, application logs, and results of other intrusion detection sensors; these data sources were referred to as an event stream.
- Used anomaly detection based on the previously published NIDES algorithms to analyze the event stream.
- Used signature detection to analyze the event stream for known examples of misuse.
- Provided an Application Programming Interface (API) to allow third-party modules and sensors to participate in the system.
- Integrated analysis from an anomaly detection engine, a signature detection engine.
- Supported hierarchical monitoring with domain NIDES monitors reporting to higher-level NIDES monitors.

269. The input stream processed by NIDES was somewhat generalized with 10 general audit record formats supported, another 90 potential types reserved, and 50 allowable user-defined audit types. The primary audit types that SRI worked with were SunOS BSM version 1, SunOS C2, and UNIX accounting records. SRI provided a small application (agen) to allow users to convert their own data sources into a format for NIDES.

270. Figure 10 shows a composite view of the NIDES system as described in *Network NIDES*. This architecture is based on what was described as implemented as of the time of this report as well as what was planned as described in Section 3 "Future Directions."

271. In the lower left portion of Figure 10 is a description of the basic NIDES monitor responsible for a particular domain (1). It would read an input stream such as network packets (2), and then analyzed that input stream with an anomaly detection engine (3) and a signature detection engine (4). The results of these analyses engines were passed to the Resolver (5), which could integrate the results, take some type of response such as alerting a user (6), or pass its analysis to hierarchically higher monitors

(7). The boxes to the right of the Domain NIDES monitor (1) are simply other Domain NIDES monitors.

272. Above the local domain NIDES monitors is a "higher-level NIDES" monitor (8) that would process the results from the lower level Domain NIDES monitors (1). This higher-level NIDES monitor would be "responsible for the network that supports all the local domains."[133]

273. New rules would be developed specifically to detect "suspicious network activity."[134] Also, the statistical component would be "easily adaptable to network intrusion detection"[135] by adapting the subjects (e.g., hosts instead of users) and measures for those subjects (e.g., "the amount of traffic originating or being received" by the subjects).[136]

274. NIDES processed any type of input stream as long as it is mapped to the NIDES audit format via an application called "agen." In addition to those data sources that were implemented for NIDES, the report also suggested input streams could consist of network packets, reports from other NIDES monitors, "a series of connection attempts," "the amount of traffic," reports from the Network Intruder Detector (NID), reports from TCP wrappers about connection attempts, reports from firewalls, SNMP, and reports from application gateways.[137]

275. Furthermore, *Network NIDES* monitors would be considered monitors under any of the parties' claim constructions.

---

[133] *Network NIDES* at 31.
[134] *Network NIDES* at 31.
[135] *Network NIDES* at 32.
[136] *Network NIDES* at 31.
[137] *Network NIDES* at 17, 32, 33.

276. In my opinion, as more fully reflected in Exhibit K to my report, the system disclosed in *Network NIDES* satisfied every limitation of the indicated claims of the patents-in-suit and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the alleged inventions claimed in the patents. Consequently, the noted claims of the patents were not novel when filed.

277. In addition, as reflect in Exhibit K, the remaining asserted claims of the patents-in-suit are invalid due to obviousness.

278. In particular, it would have been obvious to combine the *Network NIDES* system with other hierarchical intrusion detection systems such as DIDS and GrIDS, and to use their correlation capabilities in *Network NIDES*. It also would have been obvious to combine *Network NIDES* with other systems analyzing network packets and different categories of network traffic data, and to perform NIDES statistical profiling on these categories.

279. In addition, to the extent there were any differences between the configuration of *Network NIDES* and the configuration described in the patents-in-suit, it would have been obvious to modify the configuration of *Network NIDES* based upon the nature of the problem to be solved. Such configuration changes would have been motivated by a desire to address the problem of detecting and responding to suspicious network activity in very large enterprise networks.

**8. Frank Feather Thesis**

280. This section covers Frank Feather's dissertation as described in his thesis "Fault Detection in an Ethernet Network via Anomaly Detectors," dated May of 1992 ("*Feather Thesis*"). This dissertation was based on Frank Feather's research at Carnegie Mellon University and extended the Performance and Anomaly Monitoring System (PAMS) developed primarily by Pamela Hayes. Hayes' work on PAMS was published

as a Senior Thesis at Chatham College, Pittsburgh, PA in May of 1987 ("Characterizing Numeric Time Series Data: Applications to Ethernet LAN Data"). The *Feather Thesis* covered work up through May of 1992. The original PAMS work was published in 1987.

281. The *Feather Thesis* describes a network monitoring system that used an algorithm that was very similar to the Haystack statistical intrusion detection algorithm. The work's primary focus was to quickly detect a wide range of network faults, including those generated by a number of malicious or intrusive activities. The system performed fault detection by applying anomaly detection to a number of measured features of the network traffic. An anomaly was determined by comparing current measurements of network activity to historical measurements for the same measurements; the historical measurements were regularly updated using an exponential aging factor like that used in NIDES. Once anomalous activity was detected, the system further refined the analysis by applying a fault feature vector to the anomaly vector in order to determine the likely underlying cause of the detected anomalous activity.

282. Feather's dissertation had the following salient features:

- Monitored network packets.
- Designed to detect faults in the network ranging from a wide range of causes, including network worms, inappropriate use of the network, mischievous users, or malicious network misuse (e.g., connecting to the network without permission).
- Extended an earlier anomaly detection system called "Performance and Anomaly Monitoring Systems" (PAMS).
- Added the Address Anomaly Detection System (AADS) to extend PAMS' network-wide analysis to host and host-to-host analysis.
- Applied profiles of past activity to determine if currently observed network activity was anomalous.
- Updated profiles using the same exponential decay approach used by SRI's NIDES algorithm.

- Applied a fault feature vector to help determine the cause of the detected anomalous activity.
- Discussed a wide range of network features to measure.
- Deployed and evaluated on an Ethernet network analyzing packets including TCP/IP packets.
- Closely resembled the Haystack intrusion detection system algorithm.
- Suggested also applying the analysis to routers, bridges, file servers, and other network components.

283. The *Feather Thesis* addressed the problem of quick detection and diagnosis of soft failures in a network before they became a noticeable problem to the users. "Soft failures" are generally characterized by degraded performance of the network. In computer security lingo, this is generally referred to as denial of service. "Hard failures" are a complete failure of a component of the network so that no packets can be delivered. Unaddressed soft failures may eventually lead to hard failures. For example, a failing component in the network may lead to intermediate packet loss (soft failure), but that failing component may eventually completely fail leading to hard failure.

284. There are many potential reasons for soft failures in the network. Some of the examples of causes for soft failure listed in the dissertation include inappropriate use of the network, temporary congestion causing delayed transmission, failed host hardware, failure of higher level protocols, mischievous users, Internet worms, or a user connecting a host to a network without permission.

285. This system described in the *Feather Thesis* measured a wide range features about the network as a whole, individual hosts, and pair-wise host communications. Examples of such measurements include packets per minute, load, broadcast packets, packet size, packet type distribution, unexpected hosts on the network, unexpected communication between pairs of hosts, and a wide range of error events.

Such error events include collisions, checksum errors, undersized packets, oversized packets, and dropped packets.

286. Once measurements for the subjects were identified, the system developed expectations of the values for these numbers; that is, it determined the normal values for each of the measurements. The normal values were determined by analyzing past behaviors for the measurements, and the expected normal values were updated regularly using an exponential smoothing of past activities such that recently measured activity were given more weight than past activities – exponential aging.

287. Once profiles of normal values were established, the system compared current measurements to historically normal values for each measurement and identified anomalous measurements.

288. To go beyond simply identifying anomalous activity, the system applied fault feature vectors to the set of anomalous features to automatically identify the likely underlying cause of the observed anomalous behavior. For example, a bridge going down would exhibit one set of anomalous features, while a host using too much memory and paging out memory to a remote disk would exhibit a different set of anomalous features. Fault feature vectors tuned to different fault types could help disambiguate the cause of the detected anomalies.

289. Figure 11 shows the basic algorithm described in this dissertation. The box on the left represents the analysis performed as part of Frank Feather's dissertation (1). At the bottom is the data source, packets, which are fed into the system (2). From the packets a description vector was computed to represent the currently observed network activity. This description vector was composed of two parts: a series of features representing system-wide performance ($X_1$, $X_2$, ..., $X_m$) (3), and a set of features for individually observed hosts or communicating pairs of hosts ($H_1$...$H_k$) (4). Together,

these measurements represented the description vector, which is a short-term profile of recent activity.

290. Next, code compared (5) the current description of the network traffic with the profiles (6) of past network traffic. The result of this comparison was an anomaly vector ($A_1$, $A_2$, ..., $A_n$) (7) describing which of the measured features were anomalous. Once anomalous activity was detected, the system tried to determine the underlying cause of the anomalous activity by applying (8) a weighted fault vector (9) for different fault types to the anomaly vector. The result of this analysis is the "matching score" (10).

291. Interestingly, the approach described in Frank Feather's dissertation is strikingly similar to Haystack Labs' Haystack Intrusion Detection System (11).[138] In Haystack, the primary data source was a system's audit records (12), and a feature vector for each user was computed (13). An anomaly detection function (14) compared the current feature vector to a profile of historical behavior (15) to determine if there were any anomalous features. The result was an anomaly vector (16) that Haystack called the bernoulli vector. To further determine what type of malicious activity might be causing the detected anomalous activity, Haystack applied (17) a weighted intrusion vector (18) for different classes of intrusive activity, and the result was a weighted intrusion score (19) for each class of intrusion. Haystack took one additional step (not shown) to determine the probability of a given session having a given weighted intrusion score or higher.

292. Under Symantec's alternative claim construction, the *Feather Thesis* would satisfy the requirement that the long-term statistical profile was an exponentially

---

[138] I am familiar with the Haystack statistical algorithm because I analyzed Haystack, including the source code, in order to perform the analysis in my 1994 paper: B. Mukherjee, L.T. Heberlein, K.N. Levitt., "Network Intrusion Detection," IEEE Network, Vol. 8 No. 3, pp. 26-41, May/June 1994.

aged probability distribution of historical values. However, Feather's short-term statistical description was not aged.

293. The *Feather Thesis* used network packets as its primary data source. The Address Anomaly Detection System (AADS) element of the dissertation, which analyzed hosts and communicating pairs of hosts, processed data coming from the commercial Ethernet-Adapter-Karte (EAK) Monitoring System (EMS) – a network monitoring card that captured the packets and collected the primary statistics.

294. Most individual measurements were analyzed using the Performance and Anomaly Monitoring System (PAMS), which was focused on analyzing univariate time series data. For any given measurement at any given time during the day, a mean and standard deviation were computed. Newly observed data measurements that exceeded the mean by some amount (e.g., more than three standard deviations above the mean) were flagged as anomalous. Furthermore, the AADS component also reported the sudden appearance of a new network address or the sudden drop out of a usually talkative network address.

295. Once anomalous activity was detected, a fault feature vector was applied to the anomaly vector in an attempt to disambiguate the underlying cause of the detected anomalous activity. The dissertation also mentioned the need to automatically correct detected faults, but it did not go into detail as to how that might be achieved.

296. In my opinion, as more fully reflected in Exhibit L to my report, the system disclosed in *Feather Thesis* satisfied every limitation of the indicated claims of the '338 patent, and enabled one of ordinary skill in the art prior to November 9, 1997 to practice the alleged inventions claimed in the '338 patent. Consequently, the noted claims of the '338 patent were not novel when filed.